IT & C

ISSN 2821 - 8469, ISSN – L 2821 - 8469, Volumul 2, Numărul 1, Martie 2023

Funcții PHP definite de utilizator în dezvoltarea WordPress

Nicolae Sfetcu

Sfetcu, Nicolae (2023), Funcții PHP definite de utilizator în dezvoltarea WordPress, *IT & C*, **2**:1, 37-40, DOI: 10.58679/IT37237, https://www.internetmobile.ro/functii-php-definite-de-utilizator-in-dezvoltarea-wordpress/

Publicat online: 10.02.2023

Funcții PHP definite de utilizator în dezvoltarea WordPress

Nicolae Sfetcu
nicolae@sfetcu.com

User-defined PHP functions in WordPress development

Abstract

PHP defines a wide range of functions as reusable blocks of statements in the core language, and many are also available in various extensions. These functions are well documented in the online PHP documentation. Custom functions can be defined by the developer. A function will not run automatically when a page loads, it can be called from anywhere and anytime within the program. PHP supports type declarations on function parameters, which are applied at runtime.

Keywords: PHP function, WordPress development, PHP, WordPress

Rezumat

PHP definește o gamă largă de funcții ca blocuri reutilizabile de instrucțiuni în limbajul de bază, și multe sunt, de asemenea, disponibile în diferite extensii. Aceste funcții sunt bine documentate în documentația online PHP. Funcțiile personalizate pot fi definite de dezvoltator. O funcție nu se va executa automat când se încarcă o pagină, ea poate fi apelată de oriunde și oricând în cadrul programului. PHP acceptă declarații de tip privind parametrii funcției, care sunt aplicate în timpul execuției.

Cuvinte cheie: funcții PHP, dezvoltarea WordPress, PHP, WordPress

IT & C, Volumul 2, Numărul 1, Martie 2023, pp. 37-40
ISSN 2821 - 8469, ISSN – L 2821 – 8469, DOI: 10.58679/IT37237
URL: https://www.internetmobile.ro/functii-php-definite-de-utilizator-in-dezvoltarea-wordpress/

O funcție poate fi definită folosind o sintaxă precum următoarea:

Exemplul #1 Pseudocod pentru a demonstra utilizările funcțiilor

```
<?php
function foo($arg_1, $arg_2, /* ..., */ $arg_n)
{
    echo "Example function.\n";
    return $retval;
}
?>
```

Orice cod PHP valid poate apărea în interiorul unei funcții, chiar și alte funcții și definiții de clasă.

Numele funcțiilor urmează aceleași reguli ca și alte etichete în PHP. Un nume valid de funcție începe cu o literă sau liniuță de subliniere, urmată de orice număr de litere, cifre sau liniuțe de subliniere. Ca expresie regulată, ar fi exprimată astfel: `^[a-zA-Z_\x80-\xff][a-zA-Z0-9_\x80-\xff]*$`.

Sfat: Consultați și Ghidul de denumire a zonei utilizatorului.

Funcțiile nu trebuie definite înainte de a fi referite, cu *excepția* cazului în care o funcție este definită condiționat, așa cum se arată în cele două exemple de mai jos.

Când o funcție este definită într-o manieră condiționată, cum ar fi cele două exemple prezentate. Definiția sa trebuie procesată *înainte* de a fi apelată.

Exemplul #2 Funcții condiționate

```
<?php

$makefoo = true;

/* Nu putem a apela foo() de aici
deoarece nu există încă,
dar putem apela bar() */

bar();

if ($makefoo) {
  function foo()
  {
    echo "I don't exist until program execution reaches me.\n";
```

```
  }
}

/* Acum putem a apela în siguranţă foo()
deoarece $makefoo a fost evaluat la adevărat */

if ($makefoo) foo();

function bar()
{
  echo "I exist immediately upon program start.\n";
}

?>
```

Exemplul #3 Funcții în cadrul funcțiilor

```
<?php
function foo()
{
  function bar()
  {
    echo "I don't exist until foo() is called.\n";
  }
}

/*  Nu putem a apela bar() încă
deoarece nu există. */

foo();

/* Acum putem apela bar(),
Procesarea lui foo()
a făcut acest lucru accesibil. */

bar();

?>
```

Toate funcțiile și clasele din PHP au sfera globală - pot fi apelate în afara unei funcții chiar dacă au fost definite în interior și invers.

PHP nu acceptă supraîncărcarea funcțiilor și nici nu este posibilă nedefinirea sau redefinirea funcțiilor declarate anterior.

Notă: Numele de funcții nu țin cont de majuscule și minuscule pentru caracterele ASCII de la `A` la `Z`, deși de obicei este o formă bună de a apela funcții așa cum apar în declarația lor.

Atât numărul variabil de argumente, cât și argumentele implicite sunt acceptate în funcții. Consultați și referințele de funcție pentru func_num_args(), func_get_arg() și func_get_args() pentru mai multe informații.

Este posibil să apelați funcții recursive în PHP.

Exemplul #4 Funcții recursive

```
<?php
function recursion($a)
{
    if ($a < 20) {
        echo "$a\n";
        recursion($a + 1);
    }
}
?>
```

Notă: Apelurile recursive de funcții/metode cu peste 100-200 de niveluri de recursivitate pot distruge stiva și pot provoca o terminare a scriptului curent. În special, recursiunea infinită este considerată o eroare de programare.